A Appendix

A.1 Implementation Details

In this section, we first introduce the dataset. We then provide the network architecture details of **MATTE**. The hyperparameter selection criteria and the training details are summarized.

A.1.1 Dataset

We use four domain datasets to train our unsupervised model, i.e., Imdb, Yelp, Amazon and Yahoo, and follow the data split provided by Li et al. [2019]. The datasets can be downloaded via https://github.com/cookielee77/DAST. The dataset details can be found in Table 1. We set the sequence length L as 25, which is the 90 percentile of the sentence length of the training dataset. Therefore, shorter sentences are padded and longer sentences are clipped. The vocabulary size is set to 10000. For the sentiment transfer, we collect 100 positive sentences in dev set based on their sentiment labels to derive s_{transfer} to flip the sentiment of the negative sentences in the test set, and vice versa. For the tense transfer, we use stanfordnlp tool [8] to identify the tense for the main verb, then collect 100 sentences of present tense in dev set to derive s_{transfer} to flip the past-tense sentences.

A.1.2 Model Architecture

We summarize our network architecture below and describe it in detail in Table 7.

Encoder: According to Xu et al. [2020], the encoder is fed with a text span $\mathbf{x}[t_1 : t_1 + m]$ extracted from the original sentence $\mathbf{x}$, where t_1 is a random word position index, m is set to 10 if $t_1 + m$ is smaller than L. H_{word} is the word embedding dimension, set to 256. H_{lstm} is the hidden states of LSTM, set to 1024. $\text{H}_{\mathbf{z}}$ is the dimension of the latent variable, set to 80. The output of the encoder is the μ, σ and $\mathbf{z}$. All of them are in shape $[\text{BS}, \text{H}_{\mathbf{z}}]$.

Decoder: Decoder is fed with the input sentence span and the generated latent variable. The final reconstructed sentence span is one timestamp delay compared to the input span, i.e., $\mathbf{x}_{t_1+1:(t_1+1)+m}$. This is generated by applying beam search to the sequence of output probability over the vocabulary V. The $\mathcal{L}_{\text{recon}}$ is to calculate the cross-entropy loss between the output probability and target sequence span.

Content Flow r_c: We apply Deep Dense Sigmoid Flow (DDSF) [Huang et al., 2018] to derive the content noise term. To incorporate the domain information, we leverage the domain embedding (after MLP) to parameterize the flow model.

Style Flow r_s: We apply spline flow [Durkan et al., 2019] to derive the noise term. Similarly, we use the conditional flow [9] with extra input. The conditional input is the combination of content variable and domain embedding. Specially, they are concatenated firstly and the result are fed into a MLP with Tanh activation to derive a attention score α The conditional input is actually the doctProdcut.

A.1.3 Training

Training details. The models were implemented in PyTorch 2.0. and Python 3.9. The VAE network is trained for a maximum of 25 epochs and a mini-batch size of 64 is used. We use early stops if the validation reconstruction loss does not decrease for three epochs. For the encoder, we use the Adam optimizer and the learning rate of 0.001. For the decoder, we use SGD with a learning rate of 0.1. For the content and style flow, we use Adam optimizer and the learning rate is 0.001. We set three different random seeds and report the average results.

Training objective. The VAE-based model is mainly trained with $\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{VAE}}$. We use a training trick to better jointly train the other three objectives. The $\mathcal{L}_{\text{sparsity}}$ could cram the information of $\mathbf{s}$ to $\mathbf{c}$, while the $\mathcal{L}_{\text{c-mask}}$ is used to prevent the ill-posed situation where $\mathbf{s}$ have zero influence. Therefore,

---

[8] `https://stanfordnlp.github.io/stanfordnlp/`

[9] The implementation refers to ConditionedSpline in `https://docs.pyro.ai/en/stable/_modules/pyro/distributions/transforms/spline.html`

Module	**Description**	**Output**
1. Encoder	Encoder for Input sentence	
Input $\mathbf{x}_{t_1:t_1+m}$	random span of sentence	
WordEmb	get word embedding	BS $\times m \times H_{\text{word}}$
Bi-LSTM	Bi-direction, 2layers	BS $\times m \times H_{\text{lstm}}$
Average Pooling	sentencet-level Rep.	BS $\times H_{\text{lstm}}$
MLP	μ and σ	BS $\times (2 \cdot H_{\mathbf{z}})$
reparameterization	Sampling	BS $\times H_{\mathbf{z}}$
2. Domain Embedding	Embedding Layer	
Input u	number of domain $\rightarrow$ u_{dim}	BS $\times$ u_{dim}
3. Content Flow r_c		
Input: $\mathbf{c}, \mathbf{u}$	domain as flow conditional input	
MLP	$\mathbf{u} \rightarrow$ conditional context	BS $\times |\text{H}_{r_c}|$
DDSF	get content noise term $\tilde{\mathbf{c}}$	BS $\times$ c_{dim}
4. Style Flow r_s		
Input: $\mathbf{c}, \mathbf{u}, \mathbf{s}$	content and domain as flow context	
Concatenate	combine $\mathbf{c}$ and $\mathbf{u}$	BS $\times (\text{c}_{\text{dim}} + \text{u}_{\text{dim}})$
MLP	Tanh activation, get attention score α	BS $\times$ c_{dim}
Element-wise Multiplication	$\alpha \odot \mathbf{c}$	BS $\times$ c_{dim}
SplineFlow	get style noise term $\tilde{\mathbf{s}}$	BS $\times$ s_{dim}
5. Decoder r_s		
Input: $\mathbf{z}$, $\mathbf{x}_{t_1:t_1+m}$	generate the next token	
Bi-LSTM	Bi-direction, 2layers	BS $\times m \times H_{\text{lstm}}$
MLP	output word probability	BS $\times m \times$ V

Table 7: **MATTE** overall architecture. DDSF is deep dense sigmoid flow, and SplineFlow is neural spline flow. m is the length of randomly extracted text span from input sentence $\mathbf{x}$.

we involve both $\mathcal{L}_{\text{sparsity}}$ and $\mathcal{L}_{\text{c-mask}}$ at the beginning of the training phrase. For $\mathcal{L}_{\text{partial}}$, it is used to sparsify the influence intersection but their separate influences change very frequently in the initial training stages. So we involve it after 3 epochs.

Computing hardware and running time. We used a machine with the following CPU specifications: AMD EPYC 7282 CPU. We use NVIDIA GeForce RTX 3090 with 24GB GPU memory. It costs approximately 190ms to run our model on this machine per epoch.

A.2 Additional Results

This section presents additional results on the hyperparameter sensitivity and the ablation studies on more datasets.

A.2.1 Hyperparameter Sensitivity

We discuss the effect of the three loss weights $\lambda_{\text{sparsity}}$,$\lambda_{\text{partial}}$ and $\lambda_{\text{c-mask}}$ in the training objective. We have performed a grid search of $\lambda_{\text{sparsity}} \in$ [1E-4,1E-3,1E-2], $\lambda_{\text{partial}} \in$ [3E-5,3E-3,3E-1] and $\lambda_{\text{c-mask}} \in$ [1E-4,1E-3,1E-2]. The best configuration is $[\lambda_{\text{sparisty}}, \lambda_{\text{partial}}, \lambda_{\text{c-mask}}]$ = [1E-4,3E-3,1E-4]. The model performance is relatively sensitive to $\lambda_{\text{sparsity}}$, so we plot the sentiment accuracy and BLEU as a function of $\lambda_{\text{sparsity}}$ in Figure 6.

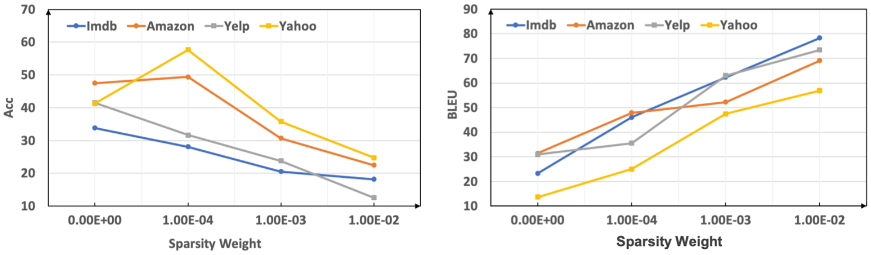

Figure 6: The Sentiment Acc (left) and BLEU (right) with different $\lambda_{\text{sparsity}}$. Among the four datasets, sentiment acc generally decreases as the $\lambda_{\text{sparsity}}$ becomes larger; BLEU increases instead. This observation aligns with our content identifiability theory. We determine the $\lambda_{\text{sparsity}}$ with the best G-score, i.e., 1E-4.

A.2.2 Ablation Results on Amazon and Yahoo Datasets

We show the ablation study of the Amazon and Yahoo datasets in Table 8. The full model achieves the best G-score and PPL on the two datasets. `CausalDep` improves the BLEU and PPL. $\lambda_{\text{sparsity}}$ greatly improves the content preservation at the cost of sentiment acc. After incorporating the $\mathcal{L}_{\text{partial}}$ and $\mathcal{L}_{\text{c-mask}}$, the sentiment acc is recovered.

	Amazon				Yahoo			
Model	Acc(↑)	BLEU(↑)	G-score(↑)	PPL(↓)	Acc(↑)	BLEU(↑)	G-score(↑)	PPL(↓)
Backbone	32.60	41.08	30.08	77.61	43.92	25.44	20.28	76.28
`Indep` [Kong et al., 2022]	**48.80**$^{\blacktriangle}$	39.50	31.76$^{\blacktriangle}$	77.95	**51.70**	23.44	21.12$^{\blacktriangle}$	56.95$^{\blacktriangle}$
`CausalDep`	33.50$^{\blacktriangle}$	45.25$^{\blacktriangle}$	32.54$^{\blacktriangle}$	66.98$^{\blacktriangle}$	41.50	31.55$^{\blacktriangle}$	21.39$^{\blacktriangle}$	64.29$^{\blacktriangle}$
:w. / $\mathcal{L}_{\texttt{sparsity}}$	27.10	**62.73**$^{\triangle}$	34.73$^{\triangle}$	**63.37**$^{\triangle}$	27.32	**48.21**$^{\triangle}$	25.16$^{\triangle}$	60.03$^{\triangle}$
:w. / $\mathcal{L}_{\texttt{partial}}$	33.10	58.54$^{\triangle}$	34.04	64.42	38.12$^{\triangle}$	41.74	28.03$^{\triangle}$	58.04$^{\triangle}$
:w. / $\mathcal{L}_{\texttt{c-mask}}$ `(Full)`	34.50$^{\triangle}$	52.25	**35.73**$^{\triangle}$	**63.37**$^{\triangle}$	38.45$^{\triangle}$	42.40$^{\triangle}$	**29.01**$^{\triangle}$	**56.12**$^{\triangle}$

Table 8: Ablation results on sentiment transfer on two domains. `CausalDep` incorporates style flow r_s to model dependency of $\mathbf{c}$ on $\mathtt{s}$, while `Indep` assumes the independence between the two variables. $^{\blacktriangle}$ marks the improvements over`Backbone`, while $^{\triangle}$ over the `CausalDep`.

A.3 Semantic Preservation Measurement by CTC score

As BLEU has limitations in capturing semantic relatedness beyond literal word-level overlap, we adopt CTC score [Deng et al., 2021] as a complementary evaluation for semantics preservation measurement. For the semantics alignment from a to b, CTC considers the matching embeddings, i.e., maximum cosine similarity of all the tokens in a with the tokens in b, and vice versa. Then, the final semantic preservation is in *F1*-style definition with one direction result as precision, and the other one as recall. The evaluation results of all the baselines and MATTE are shown in Table 9. The CTC score still favours Optimus and MATTE, with most inferior results on β-VAE, which are similar trends under the BLEU evaluation schema. Admittedly, the CTC score differences are less discriminative than BLEU – this phoneme is also observed in Liu et al. [2022].

A.4 Diversity measurements for generated sentences

To further demonstrate the generation degradation issue–generate oversimplified and repetitious sentences, we use diversity-2 [Li et al., 2016], the ratio of distinct two-grams in all the two-grams in the generated sentences to evaluate the transferred sentences. The diversity-2 for original sentences is also included for better comparison. The results in Table 10 show that all the other methods except for β-VAE generated sentences with similar diversity-2 as the original sentences, but the sentences generated by β-VAE have much lower diversity than the original ones.

	IMDB	Yelp	Amazon	Yahoo
BGST	**0.468**	0.458	**0.472**	**0.458**
β-VAE	0.436	0.433	0.433	0.413
JointTrain	0.456	0.462	0.455	0.437
CPVAE	0.462	0.463	0.461	0.443
GPT2-FT	0.459	0.459	0.458	0.448
Optimus	**0.465**	**0.468**	0.465	0.446
Matte	**0.465**	**0.464**	**0.466**	**0.452**

Table 9: CTC score, a complementary evaluation for semantics preservation. β-VAE displays the least impressive performance, and Optimus and Matte exhibit the overall best results.

Dataset	IMDB (0.34)	Yelp (0.63)	Amazon (0.64)	Yahoo(0.44)
β-VAE	0.11	0.46	0.37	0.22
JointTrain	0.21	0.59	0.56	0.37
CPVAE	0.32	0.59	0.57	0.45
MATTE	0.32	0.62	0.61	0.45

Table 10: Diversity-2 for the transferred sentences. Diversity for the original sentences is included in the bracket for comparison. β-VAE has significantly fewer distinct 2-gram than original datasets. This results are consistent with evaluation results on BLEU.

A.5 Proof for Theorem 1

The original Assumption 1 and Theorem 1 are copied below for reference.

Assumption 1 (Content identification)**.**

i. g is smooth and invertible and its inverse g^{-1} is also smooth.

ii. For all $i \in \{1, \dots, d_x\}$, there exist $\{\mathbf{z}^{(\ell)}\}_{\ell=1}^{|\mathcal{G}_{i,:}|}$ and $\mathbf{T} \in \mathcal{T}$, such that span$(\{\mathbf{J}_g(\mathbf{z}^{(\ell)})_{i,:}\}_{\ell=1}^{|\mathcal{G}_{i,:}|}) = \mathbb{R}^{d_z}_{\mathcal{G}_{i,:}}$ and $[\mathbf{J}_g(\mathbf{z}^{(\ell)})\mathbf{T}]_{i,:} \in \mathbb{R}^{d_z}_{\hat{\mathcal{G}}_{i,:}}$.

iii. For every pair (c_{j_c}, s_{j_s}) with $j_c \in [d_c]$ and $j_s \in \{d_c + 1, \dots, d_z\}$, the influence of s_{j_s} is sparser than that of c_{j_c}, i.e., $\|\mathcal{G}_{:,j_c}\|_0 > \|\mathcal{G}_{:,j_s}\|_0$.

Theorem 1. *We assume the data-generating process in Equation 1 with Assumption 1. If for given dimensions (d_c, d_s), a generative model $(p_{\hat{c}}, p_{\hat{s}|\hat{c}}, \hat{g})$ follows the same generating process and achieves the following objective:*

$$\arg\min_{p_{\hat{c}}, p_{\hat{s}}, \hat{g}} \sum_{j_{\hat{s}} \in \{d_c+1, \dots, d_z\}} \left\| \hat{\mathcal{G}}_{:,j_{\hat{s}}} \right\|_0 \quad \textit{subject to: } p_{\hat{\mathbf{x}}}(\mathbf{x}) = p_{\mathbf{x}}(\mathbf{x}),\ \forall \mathbf{x} \in \mathcal{X}, \tag{3}$$

then the estimated variable $\hat{\mathbf{c}}$ is an one-to-one mapping of the true variable $\mathbf{c}$. That is, there exists an invertible function $h_c(\cdot)$ such that $\hat{\mathbf{c}} = h_c(\mathbf{c})$.

Proof. We first define the notation $\mathbf{z} = [\mathbf{c}, \mathbf{s}]$ and the indeterminacy function:

$$h := \hat{g}^{-1} \circ g,$$

which is an invertible function $h : \mathcal{Z} \to \hat{\mathcal{Z}}$ as g is invertible by Assumption 1-i.. According to the chain rule, we have the following relation among the Jacobian matrices:

$$\mathbf{J}_{\hat{g}}(\hat{\mathbf{z}}) = \mathbf{J}_g(\mathbf{z})\mathbf{J}_h^{-1}(\mathbf{z}). \tag{8}$$

We define the support notations as follows:

$$\begin{aligned} \mathcal{G} &:= \text{supp}(\mathbf{J}_g(\mathbf{z})), \\ \hat{\mathcal{G}} &:= \text{supp}(\mathbf{J}_{\hat{g}}(\hat{\mathbf{z}})), \\ \mathcal{T} &:= \text{supp}(\mathbf{J}_h^{-1}(\mathbf{z})). \end{aligned}$$

In the following, we will show that $(j_c, j_{\hat{s}}) \notin \mathcal{T}$ for any $j_c \in \{1, \ldots, d_c\}$ and $j_{\hat{s}} \in \{d_c + 1, \ldots, d_c + d_s\}$. That is, $[\mathbf{J}_h^{-1}(\mathbf{z})]_{j_c, j_{\hat{s}}} = 0$, for any $j_c \in \{1, \ldots, d_c\}$ and $j_{\hat{s}} \in \{d_c + 1, \ldots, d_c + d_s\}$, which implies that $\mathbf{c}$ is not influenced by $\hat{\mathbf{s}}$.

Because of Assumption 1-ii., for any $i \in \{1, \ldots, d_{v_1} + d_{v_2}\}$, there exists $\{\mathbf{z}^{(\ell)}\}_{\ell=1}^{|\mathcal{G}_{i,:}|}$, such that $\operatorname{span}(\{\mathbf{J}_g(\mathbf{z}^{(\ell)})_{i,:}\}_{\ell=1}^{|\mathcal{G}_{i,:}|}) = \mathbb{R}^{d_z}_{\mathcal{G}_{i,:}}$.

Since $\{\mathbf{J}_g(\mathbf{z}^{(\ell)})_{i,:}\}_{\ell=1}^{|\mathcal{G}_{i,:}|}$ forms a basis of $\mathbb{R}^{d_z}_{\mathcal{G}_{i,:}}$, for any $j_0 \in \mathcal{G}_{i,:}$, we can express canonical basis vector $\mathbf{e}_{j_0} \in \mathbb{R}^{d_z}_{\mathcal{G}_{i,:}}$ as:

$$\mathbf{e}_{j_0} = \sum_{\ell \in \mathcal{G}_{i,:}} \alpha_\ell \cdot \mathbf{J}_g(\mathbf{z}^{(\ell)})_{i,:}, \tag{9}$$

where $\alpha_\ell \in \mathbb{R}$ is a coefficient.

Also, following Assumption 1-ii., there exists a deterministic matrix $\mathbf{T}$ where $\mathbf{T}_{j_1,j_2} \neq 0$ iff $(j_1, j_2) \in \mathcal{T}$ and

$$\mathbf{T}_{j_0,:} = \mathbf{e}_{j_0}^\top \mathbf{T} = \sum_{\ell \in \mathcal{G}_{i,:}} \alpha_\ell \cdot \mathbf{J}_g(\mathbf{z}^{(\ell)})_{i,:} \mathbf{T} \in \mathbb{R}^{d_z}_{\hat{\mathcal{G}}_{i,:}}, \tag{10}$$

where $\in$ is because each element in the summation belongs to $\mathbb{R}^{d_z}_{\hat{\mathcal{G}}_{i,:}}$.

Therefore,

$$\forall j \in \mathcal{G}_{i,:}, \mathbf{T}_{j,:} \in \mathbb{R}^{d_z}_{\hat{\mathcal{G}}_{i,:}}.$$

Equivalently, we have:

$$\forall (i,j) \in \mathcal{G}, \quad \{i\} \times \mathcal{T}_{j,:} \subset \hat{\mathcal{G}}. \tag{11}$$

As both $\mathbf{J}_g$ and $\mathbf{J}_{\hat{g}}$ are invertible, $\mathbf{J}_h(\mathbf{z})$ is an invertible matrix and thus has a non-zero determinant. Expressing $\mathbf{J}_h(\mathbf{z})$ with the Leibniz formulae gives:

$$\det(\mathbf{J}_h(\mathbf{z})) = \sum_{\sigma \in \mathcal{P}_{d_z}} \left(\operatorname{sign}(\sigma) \prod_{j=1}^{d_z} \mathbf{J}_g(\mathbf{z})_{\sigma(j),j} \right) \neq 0, \tag{12}$$

where $\mathcal{P}_{d_z}$ is the set of all d_z-permutations.

Equation 12 indicates that there exists $\sigma \in \mathcal{P}_{d_z}$, such that $\prod_{j=1}^{d_z} \mathbf{J}_g(\mathbf{z})_{\sigma(j),j} \neq 0$. Equivalently, we have

$$\forall j \in [d_z], \; (\sigma(j), j) \in \mathcal{T}. \tag{13}$$

Therefore, for a specific $j_{\hat{s}} \in \{d_c + 1, \ldots, d_z\}$, it follows that $(\sigma(j_{\hat{s}}), j_{\hat{s}}) \in \mathcal{T}$. Further, Equation 11 shows that for any $i_x \in [d_x]$, s.t., $(i_x, \sigma(j_{\hat{s}})) \in \mathcal{G}$, we have $\{i_x\} \times \mathcal{T}_{\sigma(j_{\hat{s}}),:} \subseteq \hat{\mathcal{G}}$. Together, it follows that

$$(i_x, \sigma(j_{\hat{s}})) \in \mathcal{G} \implies (i_x, j_{\hat{s}}) \in \hat{\mathcal{G}}. \tag{14}$$

Equation 14 suggests that the column $\sigma(j_{\hat{s}})$ of the true generating function support $\mathcal{G}$ is included in the column $j_{\hat{s}}$ of the estimated generating function support $\hat{\mathcal{G}}$. Together with Assumption 1-iii., it follows that

$$\sum_{j_{\hat{s}} \in \{d_c+1,\ldots,d_z\}} \left\| \hat{\mathcal{G}}_{:,j_{\hat{s}}} \right\|_0 \geq \sum_{j_s \in \{d_c+1,\ldots,d_z\}} \left\| \mathcal{G}_{:,j_s} \right\|_0, \tag{15}$$

where the permutation $\sigma(\cdot)$ connects the indices of $\mathbf{s}$ and those of $\hat{\mathbf{s}}$. We note that Equation 15 is a lower-bound of the objective Equation 3, which can be attained by a minimizer $\hat{g} = g$.

In the following, we show by contradiction that the support of $\mathbf{J}_h^{-1}(\mathbf{z})$ does not contain $(j_c, j_{\hat{s}})$, for any $j_c \in [d_c]$ and any $j_{\hat{s}} \in \{d_c + 1, \ldots, d_c\}$, i.e., $(j_c, j_{\hat{s}}) \notin \mathcal{T}$.

We suppose that a specific $(j'_c, j'_{\hat{s}}) \in \mathcal{T}$, where $j'_c \in [d_c]$ and any $j'_{\hat{s}} \in \{d_c+1, \dots, d_c\}$. We note that the argument for Equation 14 also applies to $(j_1, j_2) \in \mathcal{T}$ for any $j_1, j_2 \in [d_z]$. Thus, we would have

$$(j'_c, j'_{\hat{s}}) \in \mathcal{T} \implies (i_x, j'_{\hat{s}}) \in \hat{\mathcal{G}}, \quad \forall i_x \in \{i \in [d_x] : (i_x, j'_c) \in \mathcal{G}\}. \tag{16}$$

It would follow that

$$\begin{aligned}
\sum_{j_{\hat{s}} \in \{d_c+1,\dots,d_z\} \setminus \{j'_{\hat{s}}\}} \left\| \hat{\mathcal{G}}_{:,j_{\hat{s}}} \right\|_0 + \left\| \hat{\mathcal{G}}_{:,j'_{\hat{s}}} \right\|_0 &\geq \sum_{j_{\hat{s}} \in \{d_c+1,\dots,d_z\} \setminus \{j'_{\hat{s}}\}} \left\| \mathcal{G}_{:,\sigma(j_{\hat{s}})} \right\|_0 + \left\| \hat{\mathcal{G}}_{:,j'_{\hat{s}}} \right\|_0 \\
&\geq \sum_{j_{\hat{s}} \in \{d_c+1,\dots,d_z\} \setminus \{j'_{\hat{s}}\}} \left\| \mathcal{G}_{:,\sigma(j_{\hat{s}})} \right\|_0 + \left\| \mathcal{G}_{:,\sigma(j'_{\hat{s}})} \cup \mathcal{G}_{:,j'_c} \right\|_0 \\
&\geq \sum_{j_{\hat{s}} \in \{d_c+1,\dots,d_z\} \setminus \{j'_{\hat{s}}\}} \left\| \mathcal{G}_{:,\sigma(j_{\hat{s}})} \right\|_0 + \left\| \mathcal{G}_{:,j'_c} \right\|_0 \\
&\underbrace{>}_{(1)} \sum_{j_s \in \{d_c+1,\dots,d_z\}} \left\| \mathcal{G}_{:,j_s} \right\|_0 ,
\end{aligned} \tag{17}$$

where the inequality (1) is due to Assumption 1-iii. that the influence of $\mathbf{c}$ on $\mathbf{x}$ is denser than that of $\mathbf{s}$.

However, as discussed above, there exists an optimizer that attains the lower-bound Equation 15. Equation 17 contradicts the minimization objective Equation 3. Therefore, $(j'_c, j'_{\hat{s}}) \notin \mathcal{T}$, for any $j'_c \in [d_c]$ and any $j'_{\hat{s}} \in \{d_c+1, \dots, d_c\}$.

As discussed above, this implies that $\mathbf{c}$ is not influenced by $\hat{\mathbf{s}}$. Further, it follows from the invertibility of $h(\cdot)$ that $[\mathbf{J}_h(\mathbf{z})]_{j_{\hat{c}}, j_s} = 0$, for any $j_{\hat{c}} \in \{1, \dots, d_c\}$ and $j_s \in \{d_c+1, \dots, d_c+d_s\}$, which implies that $\hat{\mathbf{c}}$ is not influenced by $\mathbf{s}$. These two conditions and the invertibility of $h(\cdot)$ imply that $\hat{\mathbf{c}}$ and $\mathbf{c}$ form a one-to-one mapping.

□

A.6 Proof for Theorem 2

The original Assumption iii. and Theorem 2 are copied below for reference.

Assumption 2 (Partially intersecting influence supports)**.** *For every pair (c_{j_c}, s_{j_s}), the supports of their influences on $\mathbf{x}$ do not fully intersect, i.e., $\|\mathcal{G}_{:,j_c} \cap \mathcal{G}_{:,j_s}\|_0 < \min\{\|\mathcal{G}_{:,j_c}\|_0, \|\mathcal{G}_{:,j_s}\|_0\}$.*

Theorem 2. *We follow the data-generating process Equation 1 and Assumption 1 and Assumption 2. We optimize the objective function in Equation 3 together with*

$$\min \sum_{(j_{\hat{c}}, j_{\hat{s}}) \in \{1,\dots,d_c\} \times \{d_c+1,\dots,d_z\}} \left\| \hat{\mathcal{G}}_{:,j_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j_{\hat{s}}} \right\|_0 . \tag{4}$$

The estimated style variable $\hat{\mathbf{s}}$ is a one-to-one mapping to the true variable $\mathbf{s}$. That is, there exists an invertible mapping $h_s(\cdot)$ between $\mathbf{s}$ and $\hat{\mathbf{s}}$, i.e., $\hat{\mathbf{s}} = h_s(\mathbf{s})$.

Proof. As shown in Section A.5, there exists a d_z-permutation $\sigma(\cdot)$ such that $\forall j \in [d_z]$, $(\sigma(j), j) \in \mathcal{T}$. Also, we have shown in Theorem 1 that $(j_c, j_{\hat{s}}) \notin \mathcal{T}$ for $j_c \in [d_c]$ and $j_{\hat{s}} \in \{d_c+1, \dots, d_z\}$, which implies that $\sigma(j_{\hat{s}}) \in \{d_c+1, \dots, d_z\}$. Thus, it follows that for any $j_{\hat{c}} \in [d_c]$, $\sigma(j_{\hat{c}}) \in [d_c]$.

In the following, we show by contradiction that $(j_s, j_{\hat{c}}) \notin \mathcal{T}$ for any $j_s \in \{d_c+1, \dots, d_z\}$ and $j_{\hat{c}} \in [d_c]$. We suppose that $(j'_s, j'_{\hat{c}}) \in \mathcal{T}$. Analogous to Equation 16, we would have that

$$(j'_s, j'_{\hat{c}}) \in \mathcal{T} \implies (i_x, j'_{\hat{c}}) \in \hat{\mathcal{G}}, \quad \forall i_x \in \{i \in [d_x] : (i_x, j'_s) \in \mathcal{G}\}. \tag{18}$$

It would follow that $\hat{\mathcal{G}}_{:,j'_{\hat{c}}} \supseteq \mathcal{G}_{:,\sigma(j'_{\hat{c}})} \cup \mathcal{G}_{:,j'_s}$. Also, to attain the objective Equation 3 in Theorem 1, we have $j'_{\hat{s}} := \sigma^{-1}(j'_s) \in \{d_c+1, \dots, d_z\}$, s.t., $\hat{\mathcal{G}}_{:,j'_{\hat{s}}} = \mathcal{G}_{:,j'_s}$. It would follow that $\hat{\mathcal{G}}_{:,j'_{\hat{c}}} \supseteq \hat{\mathcal{G}}_{:,j'_{\hat{s}}}$.

Further, we would have

$$\left\| \hat{\mathcal{G}}_{:,j'_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j'_{\hat{s}}} \right\|_0 = \left\| \hat{\mathcal{G}}_{:,j'_{\hat{s}}} \right\|_0 = \left\| \mathcal{G}_{:,j'_s} \right\|_0 \underbrace{>}_{(2)} \left\| \mathcal{G}_{:,\sigma(j'_{\hat{c}})} \cap \mathcal{G}_{:,\sigma(j'_{\hat{s}})} \right\|_0 , \tag{19}$$

where (2) is due to Assumption 2.

We note that the lower-bound for Equation 4 is

$$\sum_{(j_{\hat{c}},j_{\hat{s}})\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}} \left\|\hat{\mathcal{G}}_{:,j_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j_{\hat{s}}}\right\|_0 \geq \sum_{(j_{\hat{c}},j_{\hat{s}})\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}} \left\|\mathcal{G}_{:,\sigma(j_{\hat{c}})} \cap \mathcal{G}_{:,\sigma(j_{\hat{s}})}\right\|_0 \tag{20}$$

$$= \sum_{(j_c,j_s)\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}} \left\|\mathcal{G}_{:,j_c} \cap \mathcal{G}_{:,j_s}\right\|_0 , \tag{21}$$

which can be achieved by $\mathcal{G} = \hat{\mathcal{G}}$. Note that the lower-bounds for both Equation 3 and Equation 4 can be attained simultaneously by $\mathcal{G} = \hat{\mathcal{G}}$. Hence, optimizing the sum of the two objectives does not alter the optimal value of either.

Applying a similar argument as that in Equation 15, we would have that

$$\begin{aligned}
&\sum_{(j_{\hat{c}},j_{\hat{s}})\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}} \left\|\hat{\mathcal{G}}_{:,j_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j_{\hat{s}}}\right\|_0 \\
&= \sum_{(j_{\hat{c}},j_{\hat{s}})\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}\setminus\{(j'_{\hat{c}},j'_{\hat{s}})\}} \left\|\hat{\mathcal{G}}_{:,j_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j_{\hat{s}}}\right\|_0 + \left\|\hat{\mathcal{G}}_{:,j'_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j'_{\hat{s}}}\right\|_0 \\
&\geq \sum_{(j_{\hat{c}},j_{\hat{s}})\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}\setminus\{(j'_{\hat{c}},j'_{\hat{s}})\}} \left\|\mathcal{G}_{:,\sigma(j_{\hat{c}})} \cap \mathcal{G}_{:,\sigma(j_{\hat{s}})}\right\|_0 + \left\|\hat{\mathcal{G}}_{:,j'_{\hat{c}}} \cap \hat{\mathcal{G}}_{:,j'_{\hat{s}}}\right\|_0 \qquad (22) \\
&\underbrace{>}_{(3)} \sum_{(j_{\hat{c}},j_{\hat{s}})\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}\setminus\{(j'_{\hat{c}},j'_{\hat{s}})\}} \left\|\mathcal{G}_{:,\sigma(j_{\hat{c}})} \cap \mathcal{G}_{:,\sigma(j_{\hat{s}})}\right\|_0 + \left\|\mathcal{G}_{:,\sigma(j'_{\hat{c}})} \cap \mathcal{G}_{:,\sigma(j'_{\hat{s}})}\right\|_0 \\
&= \sum_{(j_c,j_s)\in\{1,\ldots,d_c\}\times\{d_c+1,\ldots,d_z\}} \left\|\mathcal{G}_{:,j_c} \cap \mathcal{G}_{:,j_s}\right\|_0 ,
\end{aligned}$$

where (3) is due to Equation 19. Hence, this was not the minimizer of Equation 4. By contradiction, we have that $(j_s, j_{\hat{c}}) \notin \mathcal{T}$ for any $j_s \in \{d_c + 1, \ldots, d_z\}$ and $j_{\hat{c}} \in [d_c]$. This implies that $\mathbf{s}$ is not influenced by $\hat{\mathbf{c}}$. Further, it follows from the invertibility of $h(\cdot)$ that $[\mathbf{J}_h(\mathbf{z})]_{j_{\hat{s}},j_c} = 0$, for any $j_{\hat{s}} \in \{d_c+1, \ldots, d_z\}$ and $j_c \in [d_c]$, which implies that $\hat{\mathbf{s}}$ is not influenced by $\mathbf{c}$. These two conditions and the invertibility of $h(\cdot)$ imply that $\hat{\mathbf{s}}$ and $\mathbf{s}$ form a one-to-one mapping.

□